WRDC-TR-90-8007
Volume VIII
Part 28

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 28 - Rapid Application Generator Unit Test Plan

F. Glandorf

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209

September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

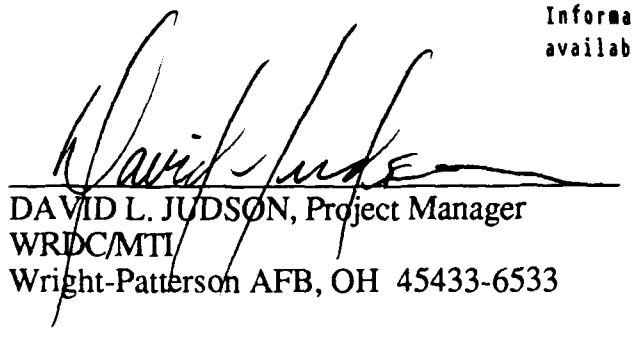
MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations



DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

FOR THE COMMANDER:

BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

If your address has changed, if you wish to be removed form our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UTP620344502	5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR- 90-8007 Vol. VIII, Part 28
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services 6b. OFFICE SYMBOL *(if applicable)*	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI
6c. ADDRESS *(City,State, and ZIP Code)* 2970 Presidential Drive Fairborn, OH 45324-6209	7b. ADDRESS *(City, State, and ZIP Code)* WPAFB, OH 45433-6533
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF 8b. OFFICE SYMBOL *(if applicable)* WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464
8c. ADDRESS *(City, State, and ZIP Code)* Wright-Patterson AFB, Ohio 45433-6533	10. SOURCE OF FUNDING NOS.
11. TITLE *(Include Security Classification)* See block 19	PROGRAM ELEMENT NO.: 78011F PROJECT NO.: 595600 TASK NO.: F95600 WORK UNIT NO.: 20950607
12. PERSONAL AUTHOR(S) Structural Dynamics Research Corporation: Glandorf, F.	
13a. TYPE OF REPORT Final Report 13b. TIME COVERED 4/1/87-12/30/90	14. DATE OF REPORT *(Yr.,Mo.,Day)* 1990 September 30 15. PAGE COUNT 38
16. SUPPLEMENTARY NOTATION WRDC/MTI Project Priority 6203	

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify block no.)*
FIELD	GROUP	SUB GR.	
1308	0905		

19. ABSTRACT *(Continue on reverse if necessary and identify block number)*

This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Rapid Application Generator Know in this document as RAP.

BLOCK 11:

INTEGRATED INFORMATION SUPPORT SYSTEM
Vol VIII -User Interface Subsystem

Part 28 - Rapid Application Generator Unit Test Plan

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson	22b. TELEPHONE NO. *(Include Area Code)* (513) 255-7371 22c. OFFICE SYMBOL WRDC/MTI

EDITION OF 1 JAN 73 IS OBSOLETE

DD FORM 1473, 83 APR

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

SUBCONTRACTOR	ROLE
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.
Structural Dynamics Research Corporation	Responsible for User Interfaces, Virtual Terminal Interface,and Network Transaction Manager design, development, implementation, and support.
Arizona State University	Responsible for test bed operations and support.

TABLE OF CONTENTS

APPENDICES

LIST OF ILLUSTRATIONS

SECTION 1

GENERAL

1.1 Purpose

This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Rapid Application Generator known in this document as RAP. The RAP is a configuration item of the Integrated Information Support System (IISS) User Interface (UI).

1.2 Project References

[1] Systran, ICAM Documentation Standards, IDS150120000C, 15 September 1983.

[2] Control Data Corporation, System Design Specification, 31 May 1988.

[3] Structural Dynamics Research Corporation, Application Generator Development Specification, DS 620344502, 31 May 1988.

[4] Structural Dynamics Research Corporation, Report Writer Unit Test Plan, UTP620344501, 31 May 1988.

[5] Structural Dynamics Research Corporation, Text Editor Unit Test Plan, UTP620344600, 31 May 1988.

[6] Structural Dynamics Research Corporation, Form Processor Unit Test Plan, UTP620344200, 31 May 1988.

[7] Structural Dynamics Research Corporation, Application Interface Unit Test Plan, UTP620344700, 31 May 1988.

[8] Structural Dynamics Research Corporation, Forms Driven Form Editor Unit Test Plan, UTP620344401, 31 May 1988.

[9] Structural Dynamics Research Corporation, User Interface Services Unit Test Plan, UTP620344100, 31 May 1988.

[10] Structural Dynamics Research Corporation, Virtual Terminal Unit Test Plan, UTP620344300, 31 May 1988.

1.3 Terms and Abbreviations

Application Definition Language: an extension of the Forms Definition Language that includes retrieval of database information and conditional actions. It is used to define interactive application programs.

Application Generator: (AG), subset of the IISS User Interface that consists of software modules that generate IISS

application code and associated form definitions based on a language input. The part of the AG that generates report programs is called the Report Writer. The part of the AG that generates interactive applications is called the Rapid Application Generator.

Application Interface: (AI), subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The AI enables applications to be hosted on computers other than the host of the User Interface.

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Attribute: field characteristic such as blinking, highlighted, black, etc. and various other combinations. Background attributes are defined for forms or windows only. Foreground attributes are defined for items. Attributes may be permanent, i.e., they remain the same unless changed by the application program, or they may be temporary, i.e., they remain in effect until the window is redisplayed.

Common Data Model: (CDM), IISS subsystem that describes common data application process formats, form definitions, etc. of the IISS and includes conceptual schema, external schemas, internal schemas, and schema transformation operators.

Communication Services: allows on host interprocess communication and inter-host communication between the various Test Bed subsystems.

Communication Subsystem: (COMM), IISS subsystem that provides communication services to the Test Bed and subsystems.

Computer Program Configuration Item: (CPCI), an aggregation of computer programs or any of their discrete portions, which satisfies an end-use function.

Conceptual Schema: (CS), the standard definition used for all data in the CDM. It is based on IDEF1 information modelling.

Cursor Position: the position of the cursor after any command is issued.

Device Drivers: (DD), software modules written to handle I/O for a specific kind of terminal. The modules map terminal specific commands and data to a neutral format. Device Drivers are part of the UI Virtual Terminal.

Display List: a list of all the open forms that are currently being processed by the FP or the user.

External Schema: (ES), an application's view of the CDM's conceptual schema.

Field: two dimensional space on a terminal screen.

Field Pointer: indicates the ITEM which contains the current cursor position.

Form: structured view which may be imposed on windows or other forms. A form is composed of fields. These fields may be defined as forms, items, and windows.

Form Definition: (FD), forms definition language after compilation. It is read at runtime by the Form Processor.

Forms Definition Language: (FDL), the language in which electronic forms are defined.

Forms Driven Form Editor: (FDFE), subset of the FE which consists of a forms driven application used to create Form Definition files interactively.

Form Editor: (FE), subset of the IISS User Interface that is used to create definitions of forms. The FE consists of the Forms Driven Form Editor and the Forms Language Compiler.

Form Hierarchy: a graphic representation of the way in which forms, items and windows are related to their parent form.

Forms Language Compiler: (FLAN), subset of the FE that consists of a batch process that accepts a series of forms definition language statements and produces form definition files as output.

Form Processor: (FP), subset of the IISS User Interface that consists of a set of callable execution time routines available to an application program for form processing.

Form processor text editor: (fpte), subset of the form Processor that consists of software modules that provide text editing capabilities to all users of applications that use the Form Processor.

IISS Function Screen: the first screen that is displayed after logon. It allows the user to specify the function he wants to access and the device type and device name on which he is working.

Integrated Information Support System: (IISS), a computing environment used to investigate, demonstrate, test the concepts and produce application for information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Item: non-decomposable area of a form in which hard-coded descriptive text may be placed and the only defined areas where user data may be input/output.

Logical Device: a conceptual device that identifies a top level window of an application. It is used to distinguish between multiple applications running simultaneously on a physical device. NOTE: a single application can have more than one logical device. To the end user this also appears as multiple applications running simultaneously.

Message: descriptive text which may be returned in the standard message line on the terminal screen. They are used to warn of errors or provide other user information.

Message Line: a line on the terminal screen that is used to display messages.

Network Transaction Manager: (NTM), IISS subsystem that performs the coordination, communication and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

Neutral Data Manipulation Language: (NDML), the command language by which the CDM is accessed for the purpose of extracting, deleting, adding, or modifying data.

Open List: a list of all the forms that are currently open for an application process.

Operating System: (OS), software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

Page: instance of forms in windows that are created whenever a form is added to a window.

Paging and Scrolling: a method which allows a form to contain more data than can be displayed with provisions for viewing any portion of the data buffer.

Physical Device: a hardware terminal.

Presentation Schema: (PS), may be equivalent to a form. It is the view presented to the user of the application.

Qualified Name: the name of a form, item or window preceded by the hierarchy path so that it is uniquely identified.

Rapid Application Generator: (RAP), part of the Application Generator that generates source code for interactive programs based on a language input.

Subform: a form that is used within another form.

Text Editor: (TE), subset of the IISS User Interface that consists of a file editor that is based on the text editing functions built into the Form Processor.

User Data: data which is either input by the user or output by the application programs to items.

User Interface: (UI), IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System (UIDS) and the User Interface Management System (UIMS).

User Interface Development System: (UIDS), collection of IISS User Interface subsystems that are used by applications programmers as they develop IISS applications. The UIDS includes the Form Editor and the Application Generator.

User Interface Management System: (UIMS), the runtime UI. It consists of the Form Processor, Virtual Terminal, Application Interface, the User Interface Services and the Text Editor.

User Interface Monitor: (UIM), part of the Form Processor that handles messaging between the NTM and the UI. It also provides authorization checks and initiates applications.

User Interface Services: (UIS), subset of the IISS User Interface that consists of a package of routines that aid users in controlling their environment. It includes message management, change password, and application definition services.

User Interface/Virtual Terminal Interface: (UI/VTI), another name for the User Interface.

Virtual Terminal: (VT), subset of the IISS User Interface that performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by the UI software which constitutes the virtual terminal definition. Specific terminals are then mapped against the virtual terminal software by specific software modules written for each type of real terminal supported.

Virtual Terminal Interface: (VTI), the callable interface to the VT.

Window: dynamic area of a terminal screen on which predefined forms may be placed at run time.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 2

DEVELOPMENT ACTIVITY

2.1 Statement of Pretest Activity

During system development, the Rapid Application Generator was tested progressively. Functionality was incrementally tested and as errors were discovered by this testing, the software was corrected.

This testing was conducted by the program developers in a manual mode. Any errors were noted by the developers and corrections to the program were then made after a testing session.

2.2 Pretest Activity Results

Testing of the RAP revealed numerous flaws which were then corrected and retesting proved successful. Testing included exceptional conditions and error conditions for the language.

SECTION 3

SYSTEM DESCRIPTION

3.1 System Description

The Rapid Application Generator is used to translate application definitions written with any text editor into programs that access data bases via the CDM and manipulate the extracted data in a way determined interactively by the user.

The Application Definition Language in which the application definitions are expressed includes the Forms Definition Language and other statement types. The syntax of the Application Definition Language accepted as input to FLAN is modelled after the Forms Definition Language and the Neutral Data Manipulation Language.

The application definition is input to the RAP which accesses metadata in the CDM to determine the schema definitions of the referenced table field values. The results are a generated C file and Cobol file and separate binary form files for each form definition within the application. The Cobol code output by the RAP is constrained to be compatible with statement forms expected by the CDM precompiler. The C file contains the control logic of the application as well as the interface to the Forms Processor software. This file calls the Cobol procedure which accesses the data in the CDM.

When the Cobol file is precompiled to translate the input SELECT statements to database calls, several procedures are generated each of which must be compiled. After compilation they are linked into the application program.

The generated application communicates via the NTM with the generated request process that in turn accesses the CDM. The course of this access is determined by the user's interaction with the application program.

The interface block diagram for the Rapid Application Generator is shown in Figure 3-1.

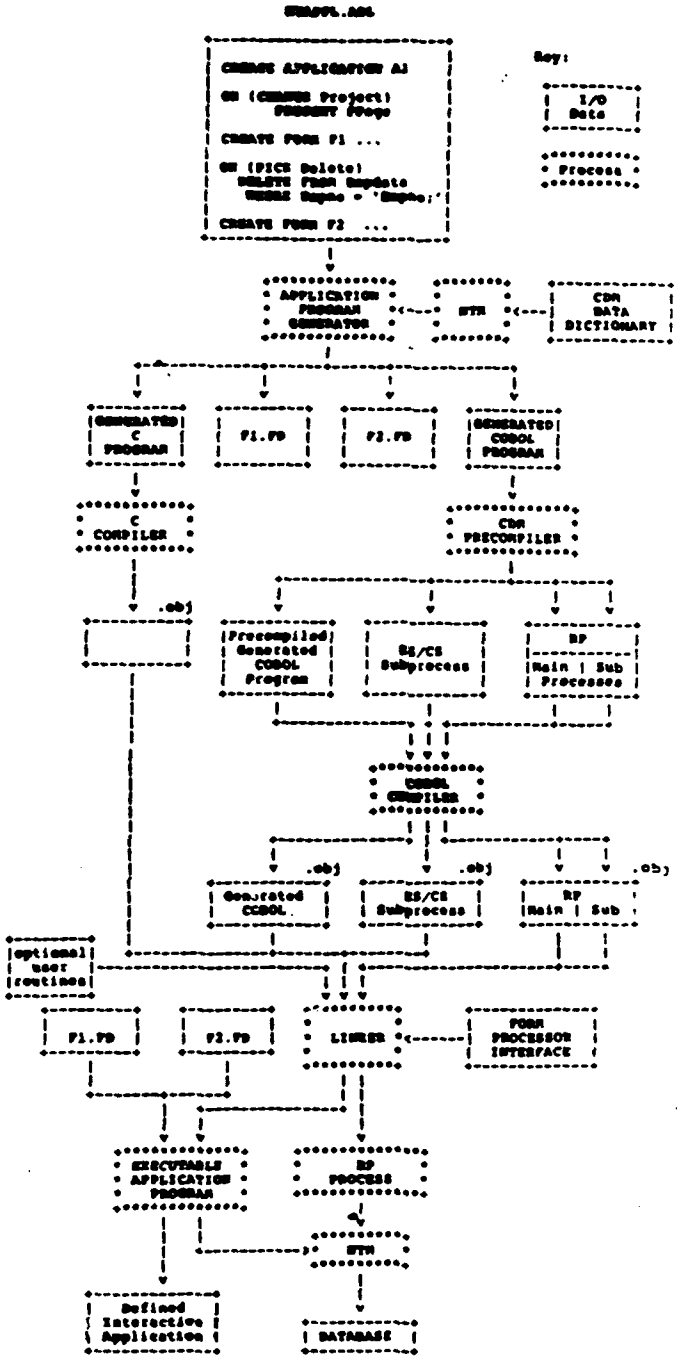


Figure 3-1 Rapid Application Generator Interfaces

3.2 Testing Schedule

The execution of the Rapid Application Generator is dependent upon the CDM and NTM subsystems of IISS and testing of the RAP must be done only after the CDM and NTM have themselves been successfully tested. Since COBOL code generated by the RAP must be precompiled, the precompiler must also be tested prior to testing of the RAP. Finally within the UI subsystem, the RAP uses the Forms Processor and Flan and therefore must be tested only after their successful tests.

3.3 First Location Testing

These tests of the RAP require the following:

Equipment: IISS Air Force Testbed VAX, terminals supported by the Virtual Terminal as listed in the IISS Terminal Operator Guide or the IISS Air Force Testbed IBM and an IBM 3270 terminal.

Support Software: the Integrated Information Support System, C compiler, and Cobol compiler.

Personnel: One integrator familiar with the IISS.

Training: The Rapid Application Generator User Manual has been provided with the current release.

Deliverables: The Rapid Application Generator subsystem of the IISS UI/VTI.

Test Materials: This test uses the files TESTAP.FDL which is in IISS Configuration Management. It can be performed using the supplied script file as outlined in this test plan if testing is performed on the VAX. On the IBM, the test must be run manually.

Security considerations: None.

3.4 Subsequent Location Testing

The requirements as listed in the previous section must be met. In subsequent tests it will not be necessary to update the UI and NTM databases if the same application names are used. This Unit Test Plan was written for the current IISS release and may need updating for future releases.

SECTION 4

SPECIFICATIONS AND EVALUATIONS

4.1 Test Specification

The following functional areas are demonstrated by the outlined tests:

Functional Requirements	Test Activity A	B	C	D	E	F	G	H	I	J	K	L	M
Startup Condition	*												
Prompt Text	*												
Present/Display/Redisplay Form	*	*	*			*	*			*			
Function Key Selection		*	*	*	*	*	*	*	*	*	*	*	*
Switching Between Functions		*						*			*		
Menu Picking		*				*	*				*		
Cursor Location		*						*	*	*	*		
Picture Specification			*										
Record Data Display			*										
Overflow of Form			*										
Database Select Operation			*										
Database Insert Operation				*									
Database Modify Operation					*								
Database Delete Operation						*							
Occurrence of Field Value				*	*	*							
Assign a Value to an Item				*	*	*							
Modify Condition				*	*	*							
Help Message									*				
Help Form										*			
Procedure Calls												*	
Exit Application													*
Chart Data Display	not yet implemented												
Icon Selection	not yet implemented												
Stretchy Lines	not yet implemented												

A - Press <ENTER> on IISS function screen.
B - Place cursor on NDML Test, press <ENTER>.
D - Press <PF5> on form TESTNDML.
C - Enter "I", "RAP_UNIT_TEST", press <ENTER>.
E - Enter "M", "MODIFY_TEST", press <ENTER>.
F - Enter "D", press <ENTER>.
G - Press <PF4> for form TESTMENU.
H - Place cursor on Test Help, press <ENTER>.
I - Place cursor on Help Message, press <ENTER>.
J - Place cursor on Help Form, press <ENTER>.
K - Place cursor on Call NTM Services, press <ENTER>.
L - Press <ENTER> on form TESTCALL.
M - Press <PF4> on form TESTMENU.

The preceding chart and the file in Appendix A show the direct correspondence between the test and the functional requirements given above.

4.2 Testing Methods and Constraints

The required input is stated for each test. This testing tests the normal mode of operations of these fucntions and does not completely exercise all the error combinations that a user of the RAP might create by faulty entry of field information. These tests have been done, however, through the normal testing done by the developer of these functions. No additional constraints are placed on this unit test besides those listed in Section 3.3 of this document.

4.3 Test Progression

The progression of testing of the RAP is fully outlined in Section 5 of this document. This progression should be followed exactly to ensure the successful testing of this IISS configuration item.

4.4 Test Evaluation

The complete Rapid Application Generator test consists of many stages each having its associated output. The first stage is the input and processing of the application definition by the generator. The outputs are the generated C and COBOL files and the binary form files named in Appendix B.

The second stage is the precompilation of the COBOL file. This shcuild be successfully produce several COBOL procedures. The names of these procedures and the names of the files containg them are constructed at generation time. The files' names as well as the success or failure of the precompilation are reported to the test evaluator in another file named according to his choice. The procedure names must be found by looking within this file. Since the CDM precompiler is not being ported to the IBM, it is necessary to move the generated COBOL code to the VAX from the IBM before it can be precompiled. This stage also compiles and links the code which has been created. The respective compilers and linker will report the success or failure of the steps comprising this stage of the test.

The third stage defines the application to the NTM tables and the UI data base. The NTM tables are updated using a text editor and the UI database is updated using the IISS utility SYSGEN.

The fourth and final stage of the test is the execution of the generated application. The success of this stage depends upon the successful operation of the NTM, the CDM and the Form Processor. This stage of the test can be considered successful if the screens output to the terminal are the same as those appearing in Appendix C. Note: the data from the queries may not be exactly the same since the contents of the CDM may change between tests, however the format of the screens should be the same.

SECTION 5

TEST PROCEDURES

5.1 Test Description

This test begins by inputting a supplied ADL source file to the Application Generator to produce form definition files and C and COBOL code. The application is then created by compiling and linking the gerated C and COBOL code as appropriate for the host system. The test also includes updating the appropriate NTM and UI database tables. The generated interactive application is then run either manually or from a supplied script file if applicable.

5.2 Test Control

As outlined, this unit test is a semiautomated test if performed on the VAX, which may be done by anyone. Once startup of the NTM has been initiated, no further operator intervention is necessary for that terminal and process. The remaining steps in stages one, two, three, and four as described in Section 4.4 above are largely manual. The comparison of the results of stage 4 is manual.

5.3 Test Procedures

The following sections document the procedures for testing the RAP on a VAX and IBM host. On a VAX host these procedures include using the Application Generator to generate the interactive application and then running the application as instructed. On an IBM host, it is not possible to use the Application Generator to generate an interactive application because the NDML Precompiler does not run there. A generated application can be run on an IBM host however, because the C code generated by the RAP is portable. The IBM test procedures demonstrate this portability.

5.3.1 VAX Host

To run the unit test plan as outlined below, one must be logged on to an IISS account. The NTM must be up and running and the UI symbolic names IISSFLIB, IISSULIB and IISSMLIB must be set properly. IISSFLIB points to the directory containing form definitions (FD files). IISSULIB points to the NTM environment directory since the Application Generator writes the FD files out to the directory you are running from and these files are used when the generated application is run. IISSMLIB points to the directory containing error messages (MSG files). This test also uses the file TESTAP.FDL which is in IISS Configuration Management and must be copied to the NTM environment directory.

If the view 'MY_VIEW' is not defined in the CDM, then the file TESTAP.DAT must be run with NDDL as:

```
$ @NDDL TESTAP.DAT
```

Below is an example of how the Rapid Application Generator may be invoked in the VAX/VMS environment for the current release. This example requires the use of two terminals. In normal usage of the RAP if the NTM is already running, only one terminal is needed. The steps are numbered sequentially with "A" and "B" following the numbers to indicate the first and second terminals. The following conventions are used in documenting the example:

- o Text in angle brackets is to be replaced with appropriate information by the user.
- o Single upper case words enclosed in angle brackets represent terminal keys (e.g. <ENTER>).
- o Text in upper case is to be entered as shown.

Stage 1 ---

1-A Logon terminal A.

2-A $ SET DEFAULT <to directory containing your NTM environment>

3-A $ @IISS This brings up the NTM.

4-B Logon terminal B.

5-B $ SET DEFAULT <to directory containing your NTM environment>

6-B $ VT100 This starts up the VT100 device driver. If the User Interface has been installed at your site with a different device driver, then this step is amended as is appropriate.

7-B Fill in the items on the IISS Logon Screen as follows:

```
Username: MORENC
Password: STANLEY
Role    : MANAGER
Press <ENTER>
```

8-B Fill in the Function item on the IISS Function Screen as follows:

```
Function: APPGENER
Press <ENTER>
```

9-B Fill in the field on the Rapid Application Generator screen as follows, where [appdir] is the directory where TESTAP.FDL resides:

```
Application Name: [appdir]TESTAP.FDL
Press <ENTER>
```

This compiles the application definition, creates the FD files, and produces both the C code (TESTAP.C) and the COBOL code (TESTAP.PRC) (PRC stands for files to be precompiled). The IISS Function screen will be displayed when the application is finished.

Press <QUIT> — This returns you to the host operating system.

Stage 2 ---

10-B $ @NDML — This invokes the NDML precompiler. Respond to the prompts as follows:

```
Name of the Logical Unit of Work:          TESTAP
Name of the Application:                   TESTAP
Name of Host Where Application Will Run:   VAX
Enter your CDM Username/Password:          CDM/CDM
Do you want old generated code deleted:    Y
Name of the PRC file:                      TESTAP
Name of the PRC file:                      <RETURN>
```

This produces the following xxxxx.TMP files:

- o Modified user modules
- o Request processor subroutine (RP-SUB) for each NDML statement found in the PRC file
- o Conceptual to external schema (CS-ES) subroutine for each NDML statement in the PRC file
- o A file called TESTAP.OUT which contains a list of the xxxxx.TMP file names and each files' contents.

11-B — Examine TESTAP.OUT for the last xxxxx.TMP file in the listing. This is a command file which will compile the CDM precompiler generated code.

```
$ @xxxxx.TMP
```

12-B — Run the following command and answer the prompts to generate the request processor main procedure (RP-MAIN) for the application.

```
$ @GENRPD
Enter Log Unit Work name:          TESTAP
Enter Oracle Username/Password:    CDM/CDM
```

13-B Type the file TESTAP.RPD. Near the bottom line is the name of the file which contains the RP-MAIN procedure. The file format will be something like xxxxx.TMP. Also record the module name, yyyyy. Compile this module using the command:

```
$ @MAINCOMP xxxxx
```

14-B The generated C file is compiled using the command:

```
$ @CCOMP TESTAP
```

15-B The application is linked using the command LNKTAP. Answer the prompts. For the name of the local RP MAIN use the name recorded from step 12-B.

```
$ @LNKTAP
Main Module:                    TESTAP
Logical  Unit of Work:          TESTAP
CDM User Name/Password:         CDM/CDM
Local Data Base:                CDM
Local Request Processor Main Module: yyyyy
```

Stage 3 ---

16-B Steps 16-B through 23-B define an application to the NTM and the UI data base. If the unit test plan must be rerun, do not perform these steps. This step compiles and links the module used to test the NTM calls.

```
$ CC REVERSE
$ @LNKAPC REVERSE
```

17-B This step updates the NTM tables.

```
$ EDIT/EDT APITBL.DAT       Insert new lines as follows:
SDTESTAPZZT1V1
SDREVERSEZT1V1

$ EDIT/EDT APTTBL.DAT       Insert new lines as follows:
TESTAPZZ9999010120001130N0
REVERSEZ9999010120001130N0
```

19-B `$ VT100`

20-B Fill in the items on the IISS Logon Screen as follows:

```
Username: MORENC
Password: STANLEY
Role    : MANAGER
Press <ENTER>
```

21-B Fill in the Function item on the IISS Function Screen as follows:

```
Function: SYSGEN
Press <ENTER>
```

22-B The SYSGEN main menu screen will be displayed.

Press <PF7> In the input field enter "TESTAP".

Press <PF7> Enter the following information to the prompts:

```
Description: Application Generator Test AP
Name:        SDTESTAPZZ
Press <ENTER>
```

23-B When the input field appears under Roles, enter a star in the field.

Press <ENTER> Application aknowledges entry.

Press <QUIT> Displays the SYSGEN main menu.

Press <QUIT> Displays the IISS Function Screen.

Press <QUIT> Returns to the host operating system.

Stage 4 ---

24-B $ VT100 NOTE that if scripting is available, Steps 24-B to 31-B may be replaced with "$ VT100 -rtestap.scp".

25-B Fill in the items on the IISS Logon Screen as follows:

```
Username: MORENC
Password: STANLEY
Role    : MANAGER
Press <ENTER>
```

26-B Fill in the Function item on the IISS Function Screen as follows:

```
Function: TESTAP
Press <ENTER>
```

27-B		The screen in figure C-1 is displayed. Place the cursor on the NDML Test field.
	Press <ENTER>	The NDML Test screen in Figure C-2 will be displayed. Enter an "I" under the Action prompt, "RAP_UNIT_TEST" under View ID, and "9999" under View Number.
	Press <ENTER>	Wait for the fields to blank.
	Press <PF5>	A query will be initiated. On the line containing the "RAP_UNIT_TEST", enter a "M" in the Action code, and "MODIFY_TEST" in the View ID.
	Press <ENTER>	Wait for the modified fields to blank.
	Press <PF5>	A query will be initiated. Note that the modified line is retrieved. On the line containing the "MODIFY_TEST", enter a "D" in the Action code.
	Press <ENTER>	Wait for the deleted fields to blank.
	Press <PF5>	A query will be initiated. Note that the deleted line is removed from the data base.
	Press <QUIT>	The test menu is displayed.
28-B		Place the cursor on the help test field.
	Press <ENTER>	The screen in figure C-3 is displayed. Place the cursor on the Help Message field.
	Press <ENTER>	The text "hello, world" is displayed in the message line. Place the cursor on the Help Form field.
	Press <ENTER>	The screen in figure C-4 is displayed.
	Press <ENTER>	This displays the help test menu.
	Press <QUIT>	This displays the test menu.
29-B		Place the cursor on the Call NTM Services field.

	Press <ENTER>	The screen in figure C-5 is displayed.
	Press <ENTER>	Note that the characters in the INPUT field are reversed and the output fields on the right are filled with data.
	Press <QUIT>	This displays the test menu.
30-B	Press <QUIT>	This terminates the application and displays the IISS function screen.
31-B	Press <QUIT>	This terminates IISS and returns you to the host operating system.
32-A	> SD > 0	This shuts down the NTM.

5.3.2 IBM Host

These procedures demonstrate the portability of the C code generated by the Application Generator. The C code is generated on a VAX host and then ported to an IBM host. It is compiled and linked to create a standalone interactive application executable. The standalone version is used so that the NTM is not needed for the test. A separate program is used to create the application test data on the IBM and then the application is run as instructed to verify the functionality.

The UI symbolic names IISSFLIB, IISSULIB, and IISSMLIB must be set properly. IISSFLIB points to the partitioned dataset containing UI form definitions (FD files). IISSULIB points to the partitioned dataset containing the application form definitions. This may be the same as IISSFLIB. Note that since this test does not include compiling the ADL source file on the IBM, you must verify that IISSULIB contains the FD files listed in Appendix B before beginning the test. IISSMLIB points to the partitioned dataset containing error messages (MSG files).

STEP 1 - Rehost the generated C code, TESTAP.C (output from step 9-B in section 5.3.1 of this document) to the IBM. Store the code as a member of any partitioned dataset. The compile JCL used in step 4 (listed in Appendix E) assumes DIISS.UI.RAP. This may be changed to whatever PDS you store the code in.

The C code must be modified since the CDM access portion of the generated code has not been ported. These edits will allow the application to read data from a predefined file and to return good return codes for the CDM access procedures. Edit the code by removing the line

```
#define GAPOPEN(x,y) (x,y,tmpfile())
```

and replace it with

```
#define GAPOPEN(x,y) (x,y,fopen("DAT(TESTAP)", "r")
TESTAP0(){ memcpy(rcode, NDMLOK, RCODE_LEN);}
TESTAP1(){ memcpy(rcode, NDMLOK, RCODE_LEN);}
TESTAP2(){ memcpy(rcode, NDMLOK, RCODE_LEN);}
TESTAP7(){ memcpy(rcode, NDMLOK, RCODE_LEN);}
```

STEP 2 - The program TSTDATA must be run to create the test data. NOTE that this program creates the test data for both a report application and an interactive application. The TSTDATA JCL compiles, links, and executes the TSTADAT program. The test data is stored in the datasets IISSCM.R23.UI.RAPDATA and IISSCM.R23.UI.RWDATA. If the Report Writer UTP has already been run, this step should not be executed for this UTP. If it has not been run, make sure that these two datasets are deleted if they exist from a previous release before executing this step. The TSTDATA source file is listed in Appendix D. From the READY prompt in TSO, submit the file IISSCM.R23.BUILD(TSTDAT) fro execution:

```
READY SUBMIT IISSCM.R23.BUILD(TSTDAT)
```

STEP 3 - From the READY prompt in TSO, submit the file IISSCM.R23.BILD(TESTAP) for execution:

```
READY SUBMIT IISSCM.R23.BILD(TESTAP)
```

This JCL compiles the test data generation program (TESTAP.C). Note that the standalone IT_routines (ITOPEN,ITSEND,etc.) are used rather than the NTM versions so that the NTM does not have to be used for the test. This JCL stores the TESTAP executable as the TESTAP member of the PDS IISSCM.R23.LOADLIB. NOTE that the condition code of * on the LINKEDT step is OK.

STEP 4 - Allocate the test data file as follows:

```
ALLOC F(DAT) ds('IISSCM.R23.LOADLIB(TESTAP)'
```

STEP 5 - Execute the generated TESTAP interactive appication as follows:

```
CALL 'IISSCM.R23.LOADLIB(TESTAP)'
```

Perform steps 14-B, 16-B and 17-B of section 5.3.1. Note that no data will be modified by the NDML actions, insert, modify and delete.

APPENDIX A

TEST APPLICATION TESTAP.FDL

The following is the file TESTAP.FDL which is the source file for the test application TESTAP. (Due to editing constraints on this document some text strings appear continued on a second line. The actual application definition under IISS Configuration Management does not contain the carriage return embedded within the string.)

```
/* NAME
 *    TESTAP - TEST APplication
 *
 * Description
 *   A test program for the rapid application generator.
 */

Create Application TESTAP

On (Startup())
   { Present TESTMENU }

On (Pick(TEST_KEY) And Cursor('TESTMENU.TESTNDML'))
   { Display Noselect TESTNDML }

   On (Pick(ACTION_KEY) And Modify('ESITEM(*).ACTION'))
      {
      On ('ESITEM(*).ACTION' = "I")
         {
         Insert Into USER_VIEW
            (VIEW_NO VIEW_ID)
         Values (0 'ESITEM(*).VIEW_ID')
      }
      On ('ESITEM(*).ACTION' = "M")
         {
         Modify USER_VIEW
            Set
               VIEW_ID = 'ESITEM(*).VIEW_ID'
            Where VIEW_NO = 'ESITEM(*).VIEW_NO'
      Set 'ESITEM(*).ACTION' = ""
      Set 'ESITEM(*).VIEW_ID' = ""
      Set 'ESITEM(*).VIEW_NO' = ""
      }
      On ('ESITEM(*).ACTION' = "D")
         {
         Delete From USER_VIEW
            Where VIEW_NO = 'ESITEM(*).VIEW_NO'
      Set 'ESITEM(*).ACTION' = ""
      Set 'ESITEM(*).VIEW_ID' = ""
      Set 'ESITEM(*).VIEW_NO' = ""
      }
      }
```

```
   On (Pick(QUERY_KEY))
      {
      Select 'ESITEM(0).VIEW_NO'
             'ESITEM(0).VIEW_ID'
         From
            (
            Select UV.VIEW_NO UV.VIEW_ID
               From USER_VIEW UV
               Where UV.VIEW_NO Not Between 100 And 2000
               Order By UV.VIEW_NO UV.VIEW_ID
            Difference
            Select UV.VIEW_NO UV.VIEW_ID
               From USER_VIEW UV
               Where UV.VIEW_NO < 100
               Order By UV.VIEW_NO UV.VIEW_ID
            Union
               (
               Select UV.VIEW_NO UV.VIEW_ID
                  From USER_VIEW UV
                  Where Not (100 >= UV.VIEW_NO Or UV.VIEW_NO >=
2000)
                  Order By UV.VIEW_NO UV.VIEW_ID
               Intersect
               Select UV.VIEW_NO UV.VIEW_ID
                  From USER_VIEW UV
                  Where UV.VIEW_NO Between 1000 And 2029
                  Order By UV.VIEW_NO UV.VIEW_ID
               )
            )
         Order By
            'ESITEM(0).VIEW_NO' Asc
         'ESITEM(0).VIEW_ID' Desc
         Present TESTNDML
      }

   On (Pick(CONT_KEY))
      { Present TESTNDML }

On (Pick(TEST_KEY) And Cursor('TESTMENU.TESTHELP'))
   { Display TESTHELP }

   On (Pick(HELP_KEY) And Cursor('TESTHELP.HELPMSG'))
      { Help "Help message displayed" }

   On (Pick(HELP_KEY) And Cursor('TESTHELP.HELPFORM'))
      { Help WORLD }

On (Pick(TEST_KEY) And Cursor('TESTMENU.TESTCALL'))
   { Display TESTCALL }

   On (Pick(CALL_KEY))
      {
      Call NSEND("SDREVERSEZ", "   ", "000000000000000", "N",
"TQ", 10,
         'TESTCALL.REVFLD', "12345")
      Call RCV("   ", "1", 'TESTCALL.SOURCE', 'TESTCALL.TYPE',
```

```
10,
          'TESTCALL.REVFLD', 'TESTCALL.RCODE', 'TESTCALL.SERNO')
      }

On (Pick(MENU_KEY))
   { Redisplay TESTMENU }

On (Pick(QUIT_AP))
   { Exit }

Create Form TESTMENU
   Keypad (TEST_KEY = 0 QUIT_AP = 4)
   Prompt Center At 1 40 "IISS Generated Application Test"
   Prompt Center At 2 40 "Test Selection Menu"

   Item TESTNDML At 4 30 Display As TABFLD Value "NDML Test"
   Item TESTHELP At 5 30 Display As TABFLD Value "Help Test"
   Item TESTCALL At 6 30 Display As TABFLD Value "Call NTM
Services"

Create Form TESTHELP
   Keypad (HELP_KEY = 0 MENU_KEY = 4)
   Prompt Center At 1 40 "IISS Generated Application Test"
   Prompt Center At 2 40 "Help Test Selection Menu"

   Item HELPMSG  At 4 30 Display As TABFLD Value "Help Message"
   Item HELPFORM At 5 30 Display As TABFLD Value "Help Form"

Create Form World
   Prompt Center At 12 40 "Hello, World"

Create Form TESTNDML
   Size 80 By 23
   Keypad (QUERY_KEY = 5 ACTION_KEY = 0 CONT_KEY = 16 MENU_KEY =
4)
   Prompt Center At 1 40 "IISS Generated Application Test"
   Prompt Center At 2 40 "NDML Actions
Insert/Select/Modify/Delete"
   Prompt At 4 2 "Action"
   Prompt At 4 10 "View ID"
   Prompt At 4 45 "View Number"
   Form ESITEM (10 V 0) At 5 1 Size 78

Create Form ESITEM
   Item ACTION At 1 4 Size 1 Display As INPUT
   Item VIEW_ID At 1 10 Size 30 Display As INPUT
   Item VIEW_NO At 1 45 Size 6 Display As OUTPUT

Create Form TESTCALL
   Keypad (CALL_KEY = 0 MENU_KEY = 4)
   Prompt Center At 1 40 "IISS Generated Application Test"
   Prompt Center At 2 40 "Call NTM Service to Reverse Characters
in Field"

   Item REVFLD At 4 15 Size 10 Display As INPUT Prompt At Left
"Input"
      value "1234567890"
```

```
   Item SOURCE At 4 40 Size 10 Display As OUTPUT Prompt At Left
"Source:"
   Item TYPE   At 5 40 Size 2  Display As OUTPUT Prompt At Left
"Msg Typ:"
   Item RCODE  At 6 40 Size 5  Display As OUTPUT Prompt At Left
"Status:"
   Item SERNO  At 7 40 Size 18 Display As OUTPUT Prompt At Left
"Ser. No:"
```

APPENDIX B

GENERATED BINARY FORM FILES

TESTMENU.FD
TESTHELP.FD
TESTNDML.FD
ESITEM.FD
TESTCALL.FD
WORLD.FD

APPENDIX C

SCREENS OUTPUT DURING TEST

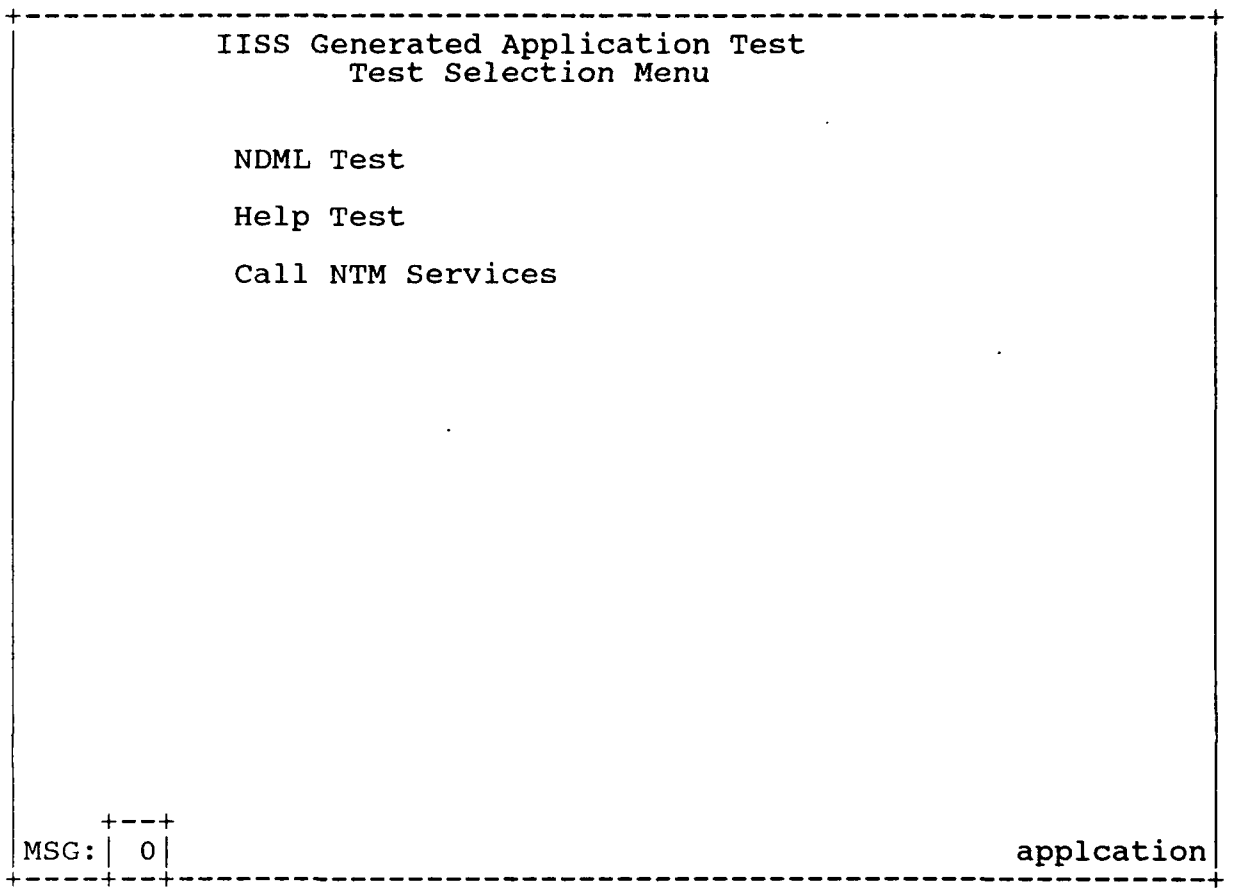


Figure C-1 TESTAP Main Menu

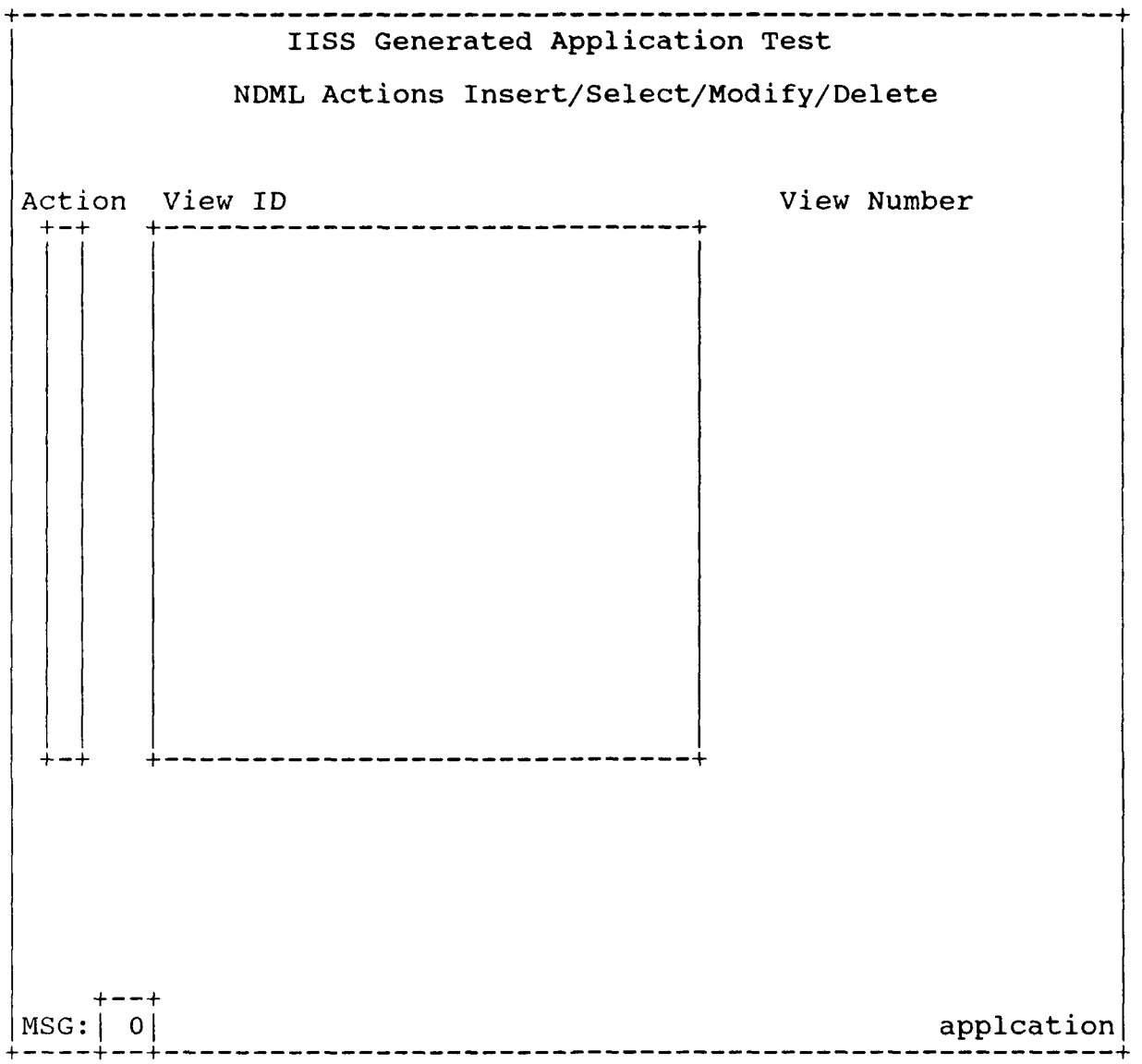


Figure C-2 NDML Test Form

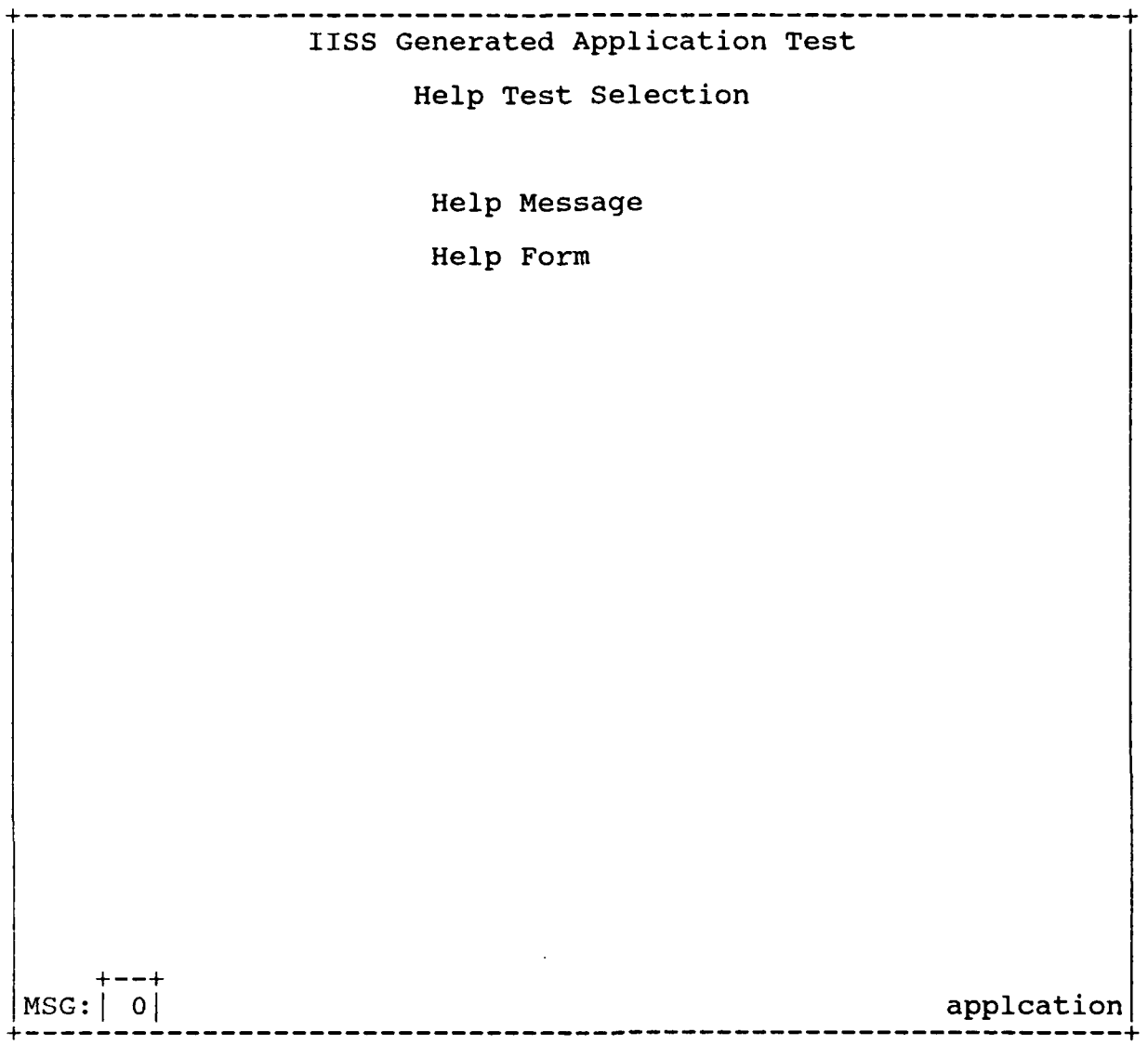


Figure C-3 Help Test Form

Figure C-4 Help Form "World"

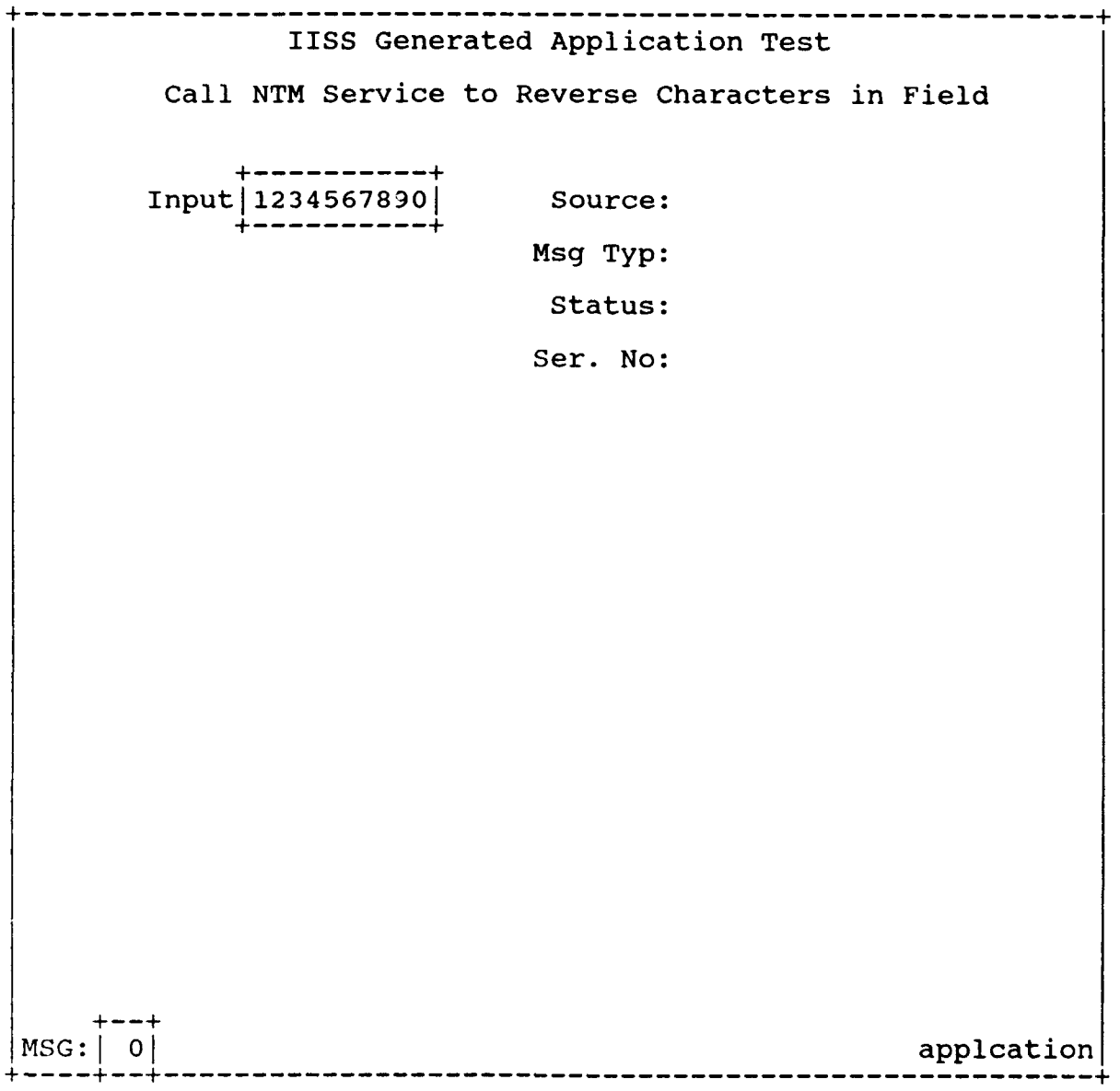


Figure C-5 NTM Service Call Test

APPENDIX D

TSTDATA SOURCE FILE

```
#include <stdtyp.h>
#include <stdio.h>


struct {
   char DBNAME[19];
   char SETID[19];
   char dummy;
   } dbr0;

struct {
   char DBID[6];
   char DBMSNAME[30];
   char dummy;
   } dbr1;

main()
{
   FILE *fptr;
   int c = 1, i = 0;

   if ((fptr = fopen("DAT", "wb")) != NULL)
   {
      sprintf(&dbr0, "%19.19s%19.19s",
         "DBNAME", "SETID");

                                       while (c && (i++ < 50))
      {
         if (i == 31)
         {
            sprintf(&dbr0, "%19.19s%19.19s",
               "123456", "SETID");
         }

         c = fwrite(&dbr0, sizeof(dbr0) - 1, 1, fptr);
      }

     fclose(fptr);

   if ((fptr = fopen("RAP", "wb")) != NULL)
   {
      i = 0;
      c = 1;

      sprintf(&dbr1, "%6.6s%30.30s",
         "12345", "RTMEMID");

      while (c && (i++ < 50))
      {
         if (i == 31)
         {
```

```
            sprintf(&dbr1, "%6.6s%30.30s",
               "12345", "RTMEMID");
         }

         c = fwrite(&dbr1, sizeof(dbr1) - 1, 1, fptr);
      }

   fclose(fptr);
}
```


★U.S. GOVERNMENT PRINTING OFFICE: 1992 - 648-127/62249